IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application

5   Applicant(s): Gn et al.
     Case: 3-3-1-1-1
     Serial No :   10/081,874
     Filing Date: February 21, 2002
     Group: 2179
10   Examiner: Mylinh T. Tran

     Title: Method and Apparatus for Generating a Graphical Interface to Enable
                    Local or Remote Access to an Application Having a Command Line
                    Interface
15   _____

TWICE CORRECTED APPEAL BRIEF

20   Mail Stop Appeal Brief - Patents
     Commissioner for Patents
     P.O. Box 1450
     Alexandria, VA 22313-1450

25   Sir:

           Appellants hereby submit this Twice Corrected Appeal Brief to conform
to the current format requirements. The original Appeal Brief was submitted on May 30,
2006 to reply to the non-final Office Action, mailed January 30, 2006. A request to
30   reinstate the appeal was submitted with the Appeal Brief. Appellant's original Appeal
Brief in an Appeal of the final rejection of claims 1-22 in the above-identified patent
application was submitted on November 10, 2005.

REAL PARTY IN INTEREST

35   The present application is assigned to Agere Systems Inc., as evidenced by
an assignment recorded on May 29, 2002 in the United States Patent and Trademark
Office at Reel 012942, Frame 0984. The assignee, Agere Systems Inc., is the real party
in interest.

40

RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

STATUS OF CLAIMS

5   Claims 1 through 22 are pending in the above-identified patent application. Claims 1-22 are rejected under 35 U.S.C. §102(b) as being anticipated by Audleman et al. (United States Patent Number 6,806,890). Claims 1, 8, 11, 12, 21, and 22 are being appealed.

10   STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the final rejection.

SUMMARY OF CLAIMED SUBJECT MATTER

The present invention is directed to generating a graphical interface (FIG
15   5: 500) for software applications having a command line interface to enable local or remote access of such software applications (page 6, lines 5-7) in a uniform manner without regard to the location of the remote application (page 4, line 20, to page 5, line 4). The location and syntax of a new software application, and any required environment settings, are specified in response to a sequence of queries (page 5, lines 5-21; page 8, 20   lines 4-26). The specifications for each software application is parsed to generate a graphical client interface (FIG. 5: 500) listing the available software applications and enabling remote access to such software applications (page 5, lines 5-21). A desired software application is selected by a user from the client interface (FIG. 1: 110) and the user specifies any necessary parameters for the selected software application (page 2, line 25   29, to page 3, line 12; page 10, lines 4-16). An input file is transferred from the client to the remote server (FIG. 1: 120) where the selected software application is located (page 5, lines 22-30). Any output or log files are returned to the client, for example, using the FTP protocol (page 6, lines 1-4). The client interface permits distributed processing through a web interface and enables software applications to be accessed and used from a 30   remote location (page 6, lines 5-12).

Independent claim 1 is directed to a method for generating a graphical interface (FIG. 5: 500) for one or more software applications having a command line interface, the method comprising the steps of: querying a user to specify properties of one or more option groups provided by each of the software applications (page 8, lines 11-

5   19); and generating a graphical user interface based on the specified properties for each of the software applications, the graphical user interface identifying each of the software applications and allowing a selected one of the software applications to be accessed (page 2, line 29, to page 3, line 17; page 5, line 5, to page 6, line 12).

Independent claim 8 is directed to a method for enabling remote access to

10   one or more software applications having a command line interface, the method comprising the steps of: querying a user to specify properties of one or more option groups provided by each of the software applications (page 8, lines 11-19); and generating a graphical user interface based on the specified properties for each of the software applications, the graphical user interface identifying each of the software

15   applications and allowing one or more clients to remotely access a selected software application (page 2, line 29, to page 3, line 17; page 5, line 5, to page 6, line 12).

In one exemplary embodiment, the remote server (FIG. 1: 120) script provides any necessary input files to the remote server (FIG. 1: 120), initiates the execution of the selected software application on the remote server (FIG. 1: 120) and

20   returns any results to the client (page 10, lines 4-28).

Independent claim 12 is directed to a system (FIG. 1: 110, 120, 300) for generating a graphical interface (FIG 5: 500) for one or more software applications having a command line interface, the system comprising: a memory (FIG. 3: 320) that stores computer-readable code; and a processor (FIG 3: 310) operatively coupled to the

25   memory (FIG. 3: 320), the processor (FIG. 3: 310) configured to implement the computer-readable code, the computer-readable code configured to: query a user to specify properties of one or more option groups provided by each of the software applications (page 8, lines 11-19); and generate a graphical user interface based on the specified properties for each of the software applications, the graphical user interface

30   identifying each of the software applications and allowing a selected one of the software applications to be accessed (page 2, line 29, to page 3, line 17; page 5, line 5, to page 6,

line 12).

Independent claim 22 is directed to an article of manufacture for generating a graphical interface (FIG. 5: 500) for one or more software applications having a command line interface, comprising: a computer readable medium having

5   computer readable code means embodied thereon, the computer readable program code means comprising: a step to query a user to specify properties of one or more option groups provided by each of the software applications (page 8, lines 11-19); and a step to generate a graphical user interface based on the specified properties for each of the software applications, the graphical user interface identifying each of the software

10   applications and allowing a selected one of the software applications to be accessed (page 2, line 29, to page 3, line 17; page 5, line 5, to page 6, line 12).

STATEMENT OF GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-22 are rejected under 35 U.S.C. §102(b) as being anticipated by

15   Audleman et al.

ARGUMENT

Independent Claims 1, 8, 12 and 22

Independent claims 1, 8, 12, and 22 were rejected under 35 U.S.C. §102(b)

20   as being anticipated by Audleman et al. In particular, the Examiner asserts that Audleman discloses querying a user to specify properties of one or more option groups provided by each of the software applications (col. 3, lines 21-20).

Applicants note that the present disclosure teaches that "the developer is then queried during step 540 to specify the properties of each *option group, i e, for the*

25   *constraints associated with a given option group, such as whether the various options within an option group can be used together and any input file requirements.*" (Page 8, lines 15-18, of the originally filed disclosure; emphasis added.) The disclosure also teaches that

the software applications have the following general

30   syntax:

Tool_name [*option 1*] [*option 2*] ..... <*filename*>
where each of these *options* further can be of one of the

4

following types {exactly one parameter; one or more than one; none or more and with or without an input file}. In this manner, the developer 210 or administrator can establish *groups and subgroups of parameters with similar properties*.

5   (Page 8, line 28, to page 9, line 4, of the originally filed disclosure; emphasis added.)

Finally, the present disclosure teaches that "the developer 210 or administrator is queried using a second interface 900, shown in FIG. 9, to specify the properties of each type of option, i.e., for the *constraints associated with a given option*

10 *group*, such as whether the various options (identified in window 950) within an option group can be used together in field 910 and any input file requirements in field 970." (Page 11, lines 21-24, of the originally filed disclosure; emphasis added.)

Once the tool is registered, "the web page 1100 allows the *user to specify the arguments for the input files for the various option groups*, as appropriate." (Page 12,

15 lines 4-6, of the originally filed disclosure; emphasis added.)

In the text cited by the Examiner, Audleman teaches that

the client 104 executes a user interface program 114 that interacts with the CS 110 to provide an operator with control over the CP's 108. The user interface program 114 receives a copy of the XML files 112

20   from the CS 110, wherein the XML files 112 represent a command syntax of the highest active release level of the command set for the CP's 108. *The user interface program 114 processes the XML files 112 to identify resource types, verbs, and keywords.* The user interface program 114 then dynamically displays a "Wizard" comprising a step-by-step series of

25   dialogs that guide the operator through the command syntax, and constructs one or more commands based on the operator selections from the dialogs.
(Col. 3, lines 18-30; emphasis added.)

30   Regarding resource types, verbs, and keywords, Audleman teaches
that

the general format of the command syntax for the CP 108 comprises the following:
        Verb Resourcetype Keyword (Parameter)
35   Each of these elements is described below:
        VERB--The verb is the first element and identifies the action to be taken. The verb can be abbreviated.
        RESOURCETYPE--The resource type is the second element and identifies the type of resource acted upon by the command. The resource
40   may comprise, for example, transactions, databases, terminals, areas,

users, etc. The resource type may support a synonym.

KEYWORD--Keywords are optional elements, depending upon the specific command. The keyword may identify resources of one or more CP's 108. The keyword may also identify a function of the command.

5

PARAMETER--A parameter identifies a defined or created value or resource, such as a database, terminal, area, user, etc. Parameters in commands must be replaced with values. Multiple parameters are separated by a comma within the parentheses.

10

(Col. 4, lines 3-22.)

Audleman, however, does not disclose or suggest *options groups* as defined in the present invention, and does not disclose or suggest *specifying properties of options groups*. Independent claims 1, 8, 12, and 22 require *querying a user to specify*

15   *properties of one or more option groups* provided by each of said software applications.

Thus, Audleman et al. do not disclose or suggest querying a user to specify properties of one or more option groups provided by each of said software applications, as required by independent claims 1, 8, 12, and 22.

20   <u>Claims 11 and 21</u>

Claims 11 and 21 are rejected under 35 U.S.C. §102(b) as being anticipated by Audleman et al. In particular, the Examiner asserts that Audleman demonstrates that the remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote

25   server and returns any results to said client (col. 2, line 55, to col. 3, line 30).

Applicants, however, could find no disclosure or suggestion by Audleman of a *remote server script* that provides any necessary input files to a remote server, and of initiating the execution of a selected software application on the remote server and returning any results to a client. Claims 11 and 21 require wherein said remote server

30   script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client.

Thus, Audleman et al. do not disclose or suggest wherein said remote server script provides any necessary input files to said remote server, initiates the

35   execution of said selected software application on said remote server and returns any

6

results to said client, as required by claims 11 and 21.

Conclusion

The rejections of the cited claims under section 102 in view of Audleman

5   et al. are therefore believed to be improper and should be withdrawn. The remaining

rejected dependent claims are believed allowable for at least the reasons identified above

with respect to the independent claims

The attention of the Examiner and the Appeal Board to this matter is

appreciated.

10   Respectfully,
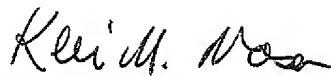
Date: March 7, 2007   Kevin M. Mason
15   Attorney for Applicant(s)
                                                                    Reg. No. 36,597
                                                                    Ryan, Mason & Lewis, LLP
                                                                    1300 Post Road, Suite 205
                                                                    Fairfield, CT 06824
20   (203) 255-6560

APPENDIX

1. A method for generating a graphical interface for one or more software applications having a command line interface, said method comprising the steps of:

5   querying a user to specify properties of one or more option groups provided by each of said software applications; and

          generating a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing a selected one of said software applications to be

10   accessed.

2. The method of claim 1, wherein said properties of each option group includes an indication of whether the various options within an option group can be used together.

15

3. The method of claim 1, wherein said properties of each option group includes an indication of any input file requirements

4. The method of claim 1, wherein said properties of each option group

20   includes a name of a corresponding software application.

5. The method of claim 1, wherein said properties of each option group includes a location of a corresponding software application

25   6. The method of claim 1, wherein said graphical user interface allows a client to access a selected software application without regard to a location of said selected software application.

7. The method of claim 1, wherein said graphical user interface presents

30   a client with only valid options for a selected software application.

8

8. A method for enabling remote access to one or more software applications having a command line interface, said method comprising the steps of:

querying a user to specify properties of one or more option groups provided by each of said software applications; and

5   generating a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing one or more clients to remotely access a selected software application

10   9. The method of claim 8, wherein a central server interacts with said one or more clients and a remote server where said selected software application is located

10. The method of claim 9, wherein said central server interacts with said one or more clients and said remote server using a remote server script.

15

11. The method of claim 10, wherein said remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client.

20   12. A system for generating a graphical interface for one or more software applications having a command line interface, said system comprising:

a memory that stores computer-readable code; and

a processor operatively coupled to said memory, said processor configured to implement said computer-readable code, said computer-readable code configured to:

25   query a user to specify properties of one or more option groups provided by each of said software applications; and

generate a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing a selected one of said software applications to be

30 accessed

13. The system of claim 12, wherein said properties of each option group includes an indication of whether the various options within an option group can be used together.

5

14. The system of claim 12, wherein said properties of each option group includes an indication of any input file requirements

15. The system of claim 12, wherein said properties of each option group includes a name of a corresponding software application.

10

16. The system of claim 12, wherein said properties of each option group includes a location of a corresponding software application.

17. The system of claim 12, wherein said graphical user interface allows a

15 client to access a selected software application without regard to a location of said selected software application.

18. The system of claim 12, wherein said graphical user interface presents a client with only valid options for a selected software application.

20

19. The system of claim 12, wherein a central server interacts with one or more clients and a remote server where said selected software application is located.

20. The system of claim 19, wherein said central server interacts with said

25 one or more clients and said remote server using a remote server script.

21. The system of claim 20, wherein said remote server script provides any necessary input files to said remote server, initiates the execution of said selected software application on said remote server and returns any results to said client.

30

22. An article of manufacture for generating a graphical interface for one or more software applications having a command line interface, comprising:

a computer readable medium having computer readable code means embodied thereon, said computer readable program code means comprising:

5   a step to query a user to specify properties of one or more option groups provided by each of said software applications; and

a step to generate a graphical user interface based on said specified properties for each of said software applications, said graphical user interface identifying each of said software applications and allowing a selected one of said software 10   applications to be accessed.

EVIDENCE APPENDIX

There is no evidence submitted pursuant to § 1.130, 1.131, or 1.132 or entered by the Examiner and relied upon by appellant.

RELATED PROCEEDINGS APPENDIX

There are no known decisions rendered by a court or the Board in any proceeding identified pursuant to paragraph (c)(1)(ii) of 37 CFR 41 37.